Lecture 14

Plan:

1) Global min cut

2) min T-odd cut.

3) Next time: matroids.

→ Mechthild Stoer

Stoer-Wagner alg for global mincut.

Setup: • Let $G=(V,E)$
• $u: E \to \mathbb{R}_{\geq 0}$

Algorithm idea:

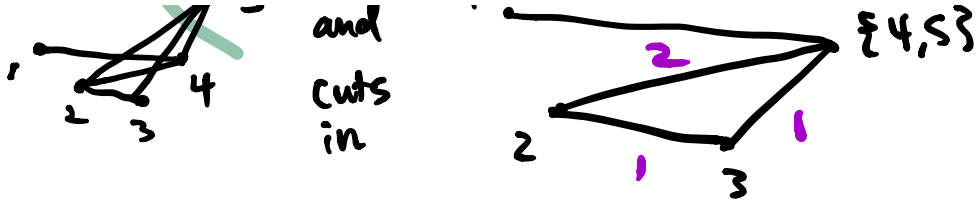
- starting with vertex, build "max adjacency ordering",

e.g. $u=1$

- Consider cut from last vertex,

e.g.

$u=1$



- **Claim:** best cut found this way is global min cut.

Def: For $A, B \subseteq V$, define

Algorithm (Stoer-Wagner)

MINCUT(G) # outputs cut.

▷ Let v_1 any vertex of G

▷ $n := |V(G)|$

create ordering

- initialize $S =$
- for $i = 2 \dots n$:
  - $v_i = \arg\min_{v \in V \setminus S}$
  - $S \leftarrow$

- if $n = 2$:
  - return
- else:
  - Get G' by shrinking
  - # recursive call
  - Let $C =$
  - return less costly of

Analysis: uses a claim.

Claim:

Claim ⇒ correctness:

- The min cut is either a min v_{n-1}-v_n cut, or not.
- If it is, claim ⇒
- If not, induction ⇒

Proof of Claim:

Let

be the ordering from alg.

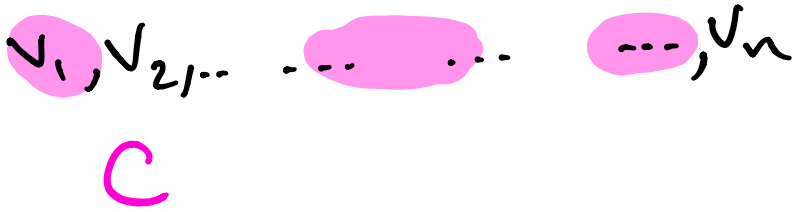
- $A_i :=$ sequence

- Consider candidate $v_{n-1}-v_n$ cut, i.e. $C \subseteq V$ s.t.

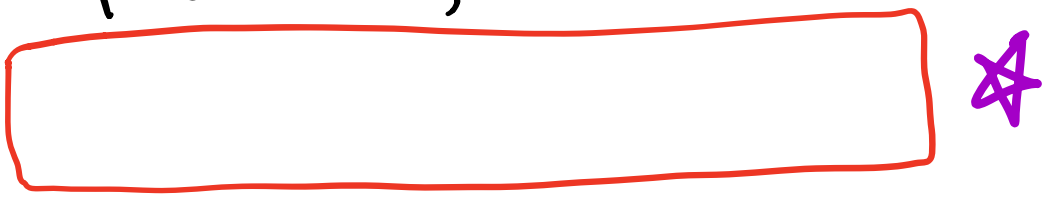
- Want to show

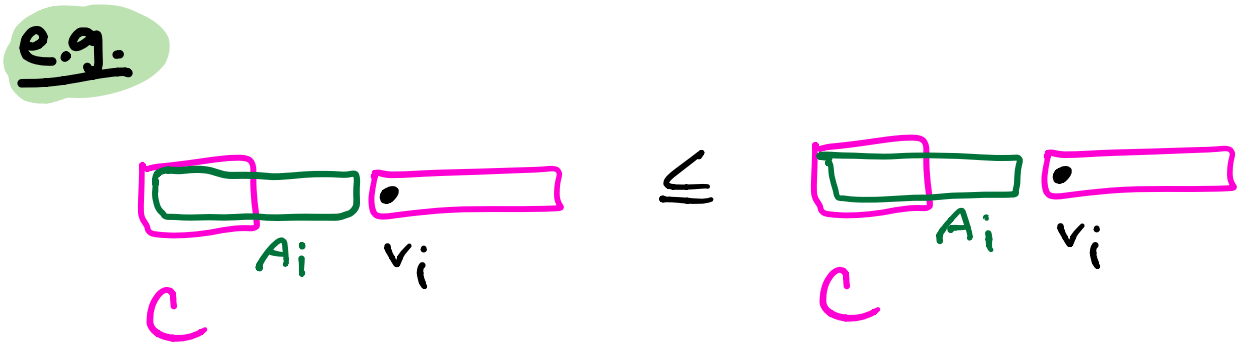
i.e.

- define v_i to be **critical** if



- **Subclaim:** Define $C_i :=$ {v_j critical, $j \le i$}





Subclaim suffices:

because v_n is critical,

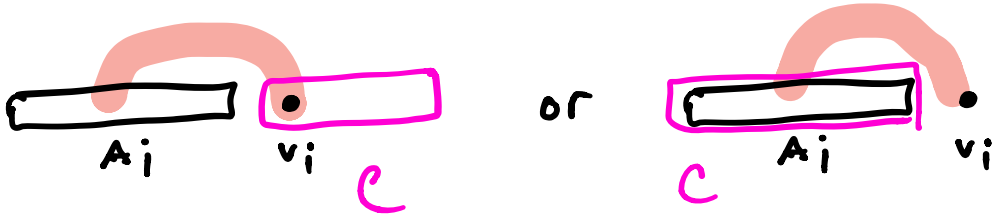
Subclaim $\Rightarrow u(\delta(A_n)) \leq u(\delta(C))$

$u(A_n : V_n)$ $\quad u(C_n : A_{n+1} \setminus C_n)$

is ✱ for $i = n$.

Proof of Subclaim:

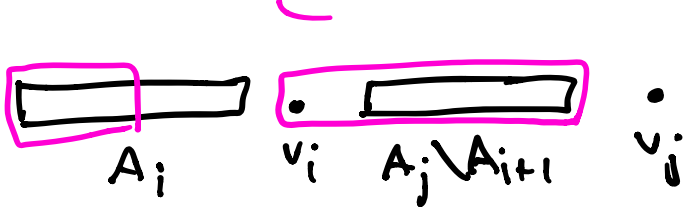
- induction on seq. of critical vertices.
- (base:) ✱ true for first critical v_i



( )

- (inductive) Assume ✡ true for critical v_i, let v_j next critical.

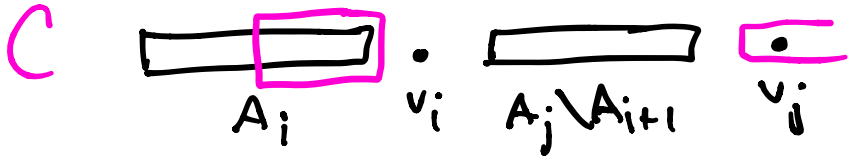
e.g.



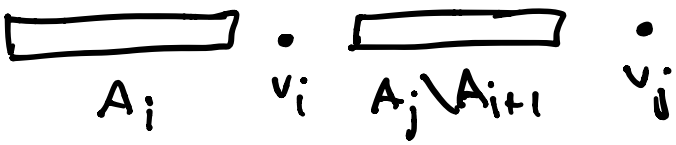
- Assume $v_i \in C$, $v_j \notin C$. (as in pic)

Is WLOG:

e.g.

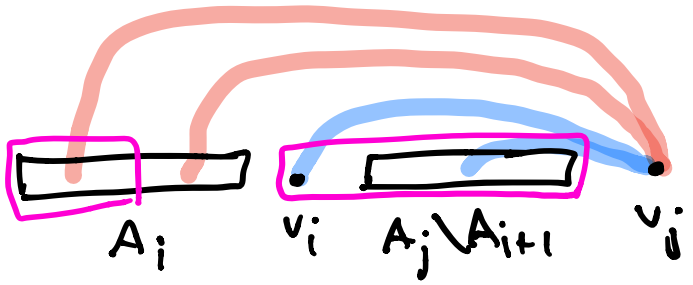


$\rightsquigarrow$

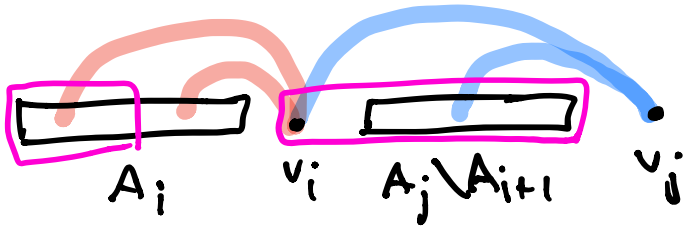


- Then

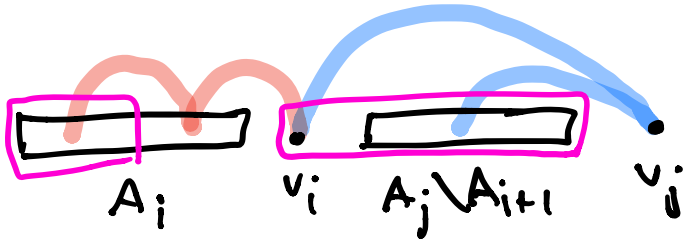
$u(A_j : \{v_j\}) =$



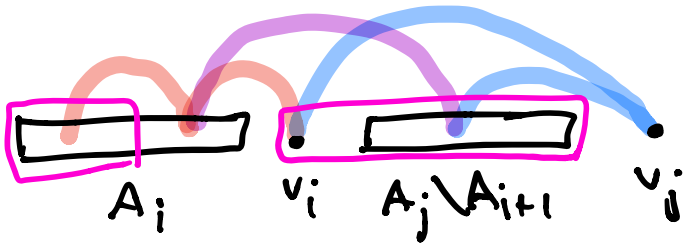
$\leq$



$\leq$



$\leq$



$\square$

Running time:

Depends how you implement ordering.

- While building ordering, must maintain list of costs

- must quickly find minimum & update new after picking v_{i+1}.
- This is what "priority queues" are for, e.g. "Fibonacci heap".
- with Fibonacci heap, can build order in time.
- total runtime
- Compare to from computing maxflows.

Submodularity

- Stoer-Wagner can be extended to minimize a more general class of functions than $S \mapsto w(\delta(S))$.

- function $f: 2^V \to \mathbb{R}$ **submodular** if [$\geq$].

- **Examples:**
  - $f(S) =$,
  - $f(S) =$ for w nonnegative, even if G directed.
  - V = food items on menu, $S \subseteq V$ meal

Submodularity equivalent to "diminishing marginal returns":

For $S \supseteq T$, $v \notin S$,

- Stoer-Wagner algorithm can be extended to minimize any symmetric ( ) submodular function.

  ↳ Queyranne '95. ( ).

Application of submodularity:

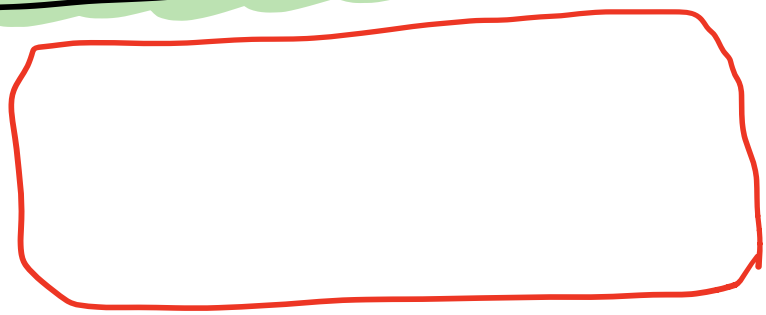
Minimum T-odd cut

- $G=(V,E)$

  $u: E \to \mathbb{R}$

  $T \subseteq V$

- **minimum T-odd cut problem:**

  $S =$ 

- Say S is **T-odd** if .

- Note:

- why do we care?

Recall

THM (Edmonds) Let

$$X = \{1_M : M \text{ matching in } G\}.$$

Then $\text{conv}(X) = P$ where

$$P = \{x : \quad \forall v \in V.$$

$$\forall S \subseteq V$$

$$|S| \text{ odd}$$

$$\forall e \in E.\}$$

- How can we quickly test if $x \in P$?

- Padberg-Rao: Can express as min odd cut problem.

- What's more, can get separating hyperplane if $x \notin P$.

- This can be used to optimize over P via ellipsoid alg, despite there being no polynomial size LP for optimizing over P (Rothvoss).

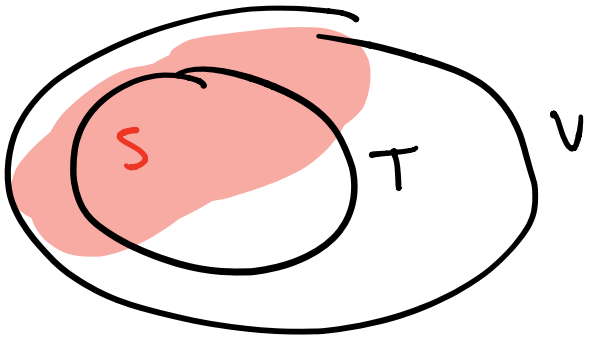
- **Today:**
  - poly. time alg. for min T-odd cut
  - crucially uses submodularity.

Algorithm ALG(G, T)

1) Find min cut among those with one vertex of T on each side:

$S =$ 

- Takes $|T|-1$ min s-t cut computations:

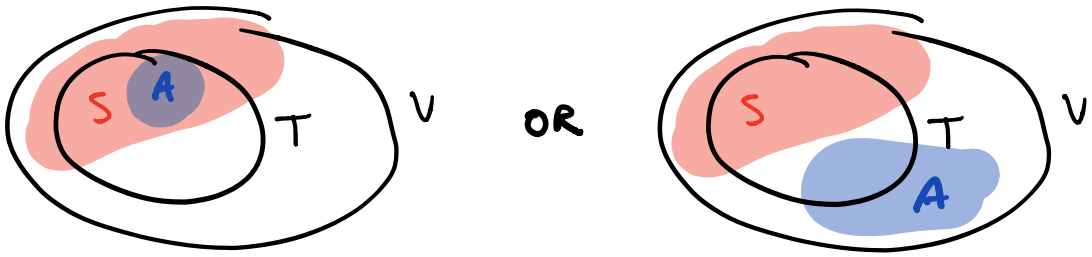


2) If S is T-odd cut, is minimum; return S.

Else: if S T-even, use

Lemma: If S as in ✡, $|S \cap T|$ even,

e.g.



Pf: after alg.

- Lemma ⇒ 2 recursive calls suffice:

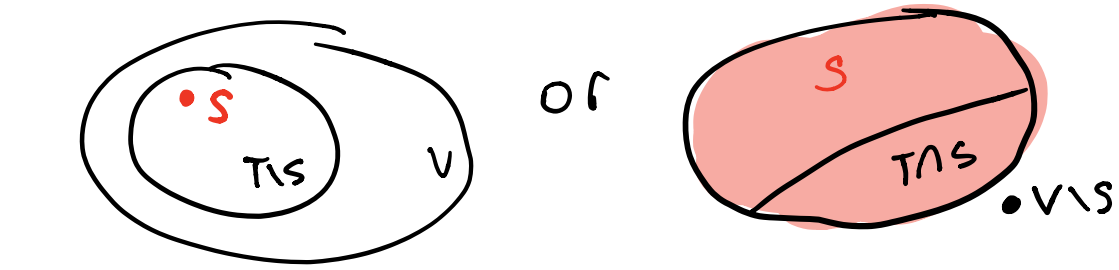
▷ $G_1 :=$

$T_1 :=$

▷ $G_2 :=$

$T_2 :=$

e.g.



Return:

min

Running time why polytime if 2 recursive calls?

$R(k) :=$

Then

a) $R(2) = T :=$

b) $R(k) \leq \max_{\substack{k_1 \geq 2 \\ k_2 \geq 2 \\ k_1 + k_2 = k}}$

By induction, $R(k) \leq$

Pf: • base: True for $k = 2$

• Inductive:

$$R(k) \leq \max_{\substack{k_1 \geq 2 \\ k_2 \geq 2 \\ k_1 + k_2 = k}}$$

$\leq$

$=$

$\leq$

Thus: algorithm is polynomial.

- Now for the lemma.
- Proof uses submodularity.

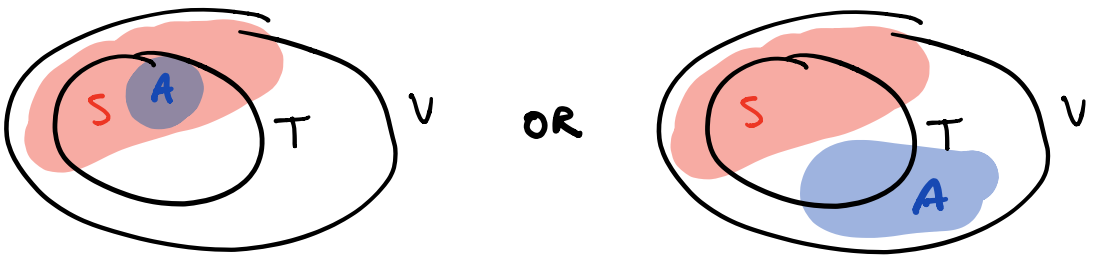
Recall:

Lemma: Let S min cut subject to $\emptyset \neq S \cap T \neq T$, $|S \cap T|$ even, then

$\exists$ min T-odd cut A with

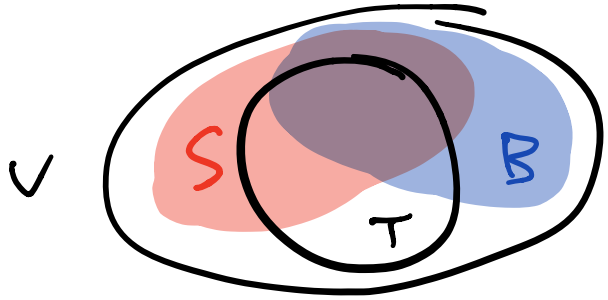
$$A \subseteq S$$

or $$A \subseteq V \setminus S.$$

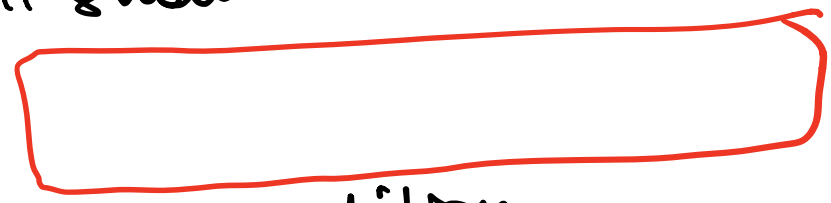


Proof

- Let B be any minimum T-odd cut.



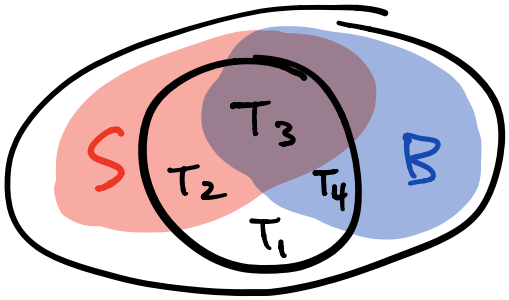
- We'll show we can take



- Make a partition.

$T_1 =$, $T_2 =$

$T_3 =$, $T_4 =$



- By definition of B, S, know all pairwise unions nonempty:

$$T_1 \cup T_4 = \emptyset$$
$$T_2 \cup T_3 \neq \emptyset$$

$$T_2 \cup T_1 \neq \emptyset$$
$$T_3 \cup T_4 \neq \emptyset$$

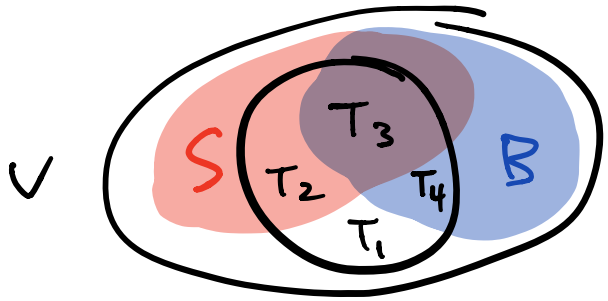
$\Rightarrow$

- By possibly replacing $B \leftarrow V \setminus B$, may assume

- submodularity of cut $\Rightarrow$

(Δ)

- As $T_1 \neq \emptyset$, $T_3 \neq \emptyset$,



- One of $S \cup B$, $S \cap B$ is T-even & the other T-odd because

Summary:

- $\triangle \Rightarrow$ whichever of SUB, SNB add has cut value $\leq$ the other $\leq$

$\Rightarrow$

$\square$